2005 A SEMESTER EXAMINATIONS

THE UNIVERSITY OF
WAIKATO
Te Whare Wānanga o Waikato

DEPARTMENT	Computer Science
PAPER TITLE	Programming Languages
TIME ALLOWED	Three Hours
NUMBER OF QUESTIONS IN PAPER	Five
NUMBER OF QUESTIONS TO BE ANSWERED	Five
VALUE OF EACH QUESTION	All questions are of equal value.
GENERAL INSTRUCTIONS	Answer ALL FIVE questions.
SPECIAL INSTRUCTIONS	Nil
CALCULATORS PERMITTED	No

1. (a) Briefly describe three of the following terms with respect to the analysis phases of a compiler:

lexical analysis

top down parser

left-most derivation

LL(1)

[6 Marks]

(b) Given the following grammar construct its First and Follow sets. The start symbol is Bexp. Nonterminals start with an uppercase letter. Terminals are bolded in lowercase.

Bexpr	→	Bterm Bexpr'
Bexpr'	→	**or** Bterm Bexpr' | **ε**
Bterm	→	Bfactor Bterm'
Bterm'	→	**and** Bfactor Bterm' | **ε**
Bfactor	→	**not** Bfactor | **(** Bexpr **)** | **true** | **false**

[9 Marks]

(c) Rewrite the following grammar so that it is suitable for predictive parsing:

Exp	→	Exp + Term | Term
Term	→	Term * Factor | Factor
Factor	→	(Exp) | **number** | **id**

[5 Marks]

2. (a) Discuss what it means for a language to be statically typed versus dynamically typed.

[4 Marks]

(b) Discuss what it means for a language to be dynamically scoped versus lexically (statically) scoped.

[4 Marks]

(c) The three different approaches used by a type checker for determining the equivalence of types are structural, declaration and name equivalence. Describe each of them using example code.

[12 Marks]

3. (a) Given the following clauses outline the effect of the cut operator (!) when Prolog is given *path (1, 4, [])*.

```
move(1,6).
move(1,8).
move(6,1).
move(6,7).
move(8,1).
move(8,3).
move(7,6).
move(7,2).
move(2,9).
move(3,4).
move(9,4).
path((Z,Z,L).
path(X,Z,L) :- move(X,Y), not (member(Y,L)), path(Y, Z, [Y|L]), ! .
```

[7 Marks]

(b) What is the closed world assumption in Prolog?

[3 Marks]

(c) Below is an example of a logic puzzle for which a solution could be found using Prolog. Using this as an example, describe how you would go about solving such logic puzzles in Prolog, i.e. discuss the methodology behind a solution. Do not provide a solution to the puzzle. Rather you need to outline an approach that could systematically be applied to solving any of these puzzles. In doing this you should discuss the predicates you need to use or implement and why you would need them. Discuss the structures and variables you need to introduce and why. You should use example code where it helps to explain your approach.

Four people ordered items from a popular mail order catalogue. They all lived in the same apartment building and all on the same floor. Each person ordered one item and each item was delivered on a different day during the same week. Write a Prolog program to determine the full name of each person, their apartment number, the day their item was delivered, and what each item was. The clues are as follows:

1. *Mr. Small, in apartment #426, didn't receive his item on Tuesday.*
2. *The man who received his item on Monday lived in apartment #437. Peter didn't get a toaster oven.*
3. *The order of deliveries was the vacuum cleaner, Mitch's item, the item for apartment #419, and the coffeemaker.*
4. *Peter Raleigh didn't receive the vacuum cleaner and didn't live in apartment #415.*
5. *Mr. Wallis, who wasn't Stan, received the toaster oven, but not on Friday.*
6. *The bookcase arrived on Thursday, but Terry, whose last name wasn't Madison, didn't receive it.*

[10 Marks]

Student Name:_____ ID Number:_____

4. (a) Briefly describe three of the following terms with respect to functional programming languages.

currying

lazy evaluation

partial evaluation of functions

referential transparency

[9 Marks]

(b) What is the type of function `somefun`?

```
somefun [] y = y
somefun (x:xs) y = x: somefun xs y
```

[2 Marks]

(c) What is the type of function *reduce*?

```
reduce (x:ys) = [y | y <- ys, x \= y]
```

[2 Marks]

(d) What is the type of *(fun.reduce)*?

[2 Marks]

(e) The arguments to a Haskell function may be polymorphic with respect to type. Discuss how, in this respect, Haskell ensures type safety.

[5 Marks]

5. (a) Java and C++ are both strongly typed languages. Discuss the restrictions that each language imposes on polymorphism to achieve this.

[10 Marks]

(b) **Either**

Write a short essay outlining which of the programming language paradigms (object oriented, procedural, functional, logic) you prefer to program in and why. You need to outline the distinguishing features of this language which in your opinion make it a good language to program in. You should also compare it in these respects, where possible, with the other programming language paradigms.

or

Choose one programming language paradigm (object oriented, procedural, functional, logic) and write a short essay discussing its contribution to the development/progression of programming languages.

or

Write an essay comparing the notion of *abstraction* in the object oriented, functional and procedural programming languages.

[10 Marks]